Enhanced template update: Application to keystroke dynamics

Paulo Henrique Pisani, Romain Giot, André C.P.L.F. de Carvalho, Ana Carolina Lorena

Enhanced Template Update: Application to Keystroke Dynamics

Paulo Henrique Pisani[a,*], Romain Giot[b], André C. P. L. F. de Carvalho[a], Ana Carolina Lorena[c]

[a] *Universidade de São Paulo, Instituto de Ciências Matemáticas e de Computação, Av. Trabalhador São Carlense, 400, São Carlos, Brazil*
[b] *Université de Bordeaux, Laboratoire Bordelais de Recherche en Informatique, UMR 5800, F-33405 Talence, France*
[c] *Universidade Federal de São Paulo, Instituto de Ciência e Tecnologia, Rua Talim, 330, São José dos Campos, Brazil*

Abstract

With the increasing number of activities being performed using computers, there is an ever growing need for advanced authentication mechanisms like biometrics. One efficient and low cost biometric modality is keystroke dynamics, which attempts to recognize users by their typing rhythm. It has been shown that the biometric features may undergo changes over time, which can reduce the predictive performance of the biometric system. Template update adapts the user model to deal with these changes and, therefore, decreases the predictive performance loss. Most of the studies in the literature only take into account samples classified as genuine to perform adaptation. This paper extends this common approach by proposing an original framework to make use of samples classified as impostors too. This new approach, named Enhanced Template Update, uses all collected unlabeled samples to support the adaptation process. According to our experimental results, this new approach can improve the predictive performance when compared to current methods depending on the scenario. Some improvements on the visualization of results over time are also proposed during the analysis performed in this study. Although the proposed approach is evaluated on keystroke dynamics, it could also be applied to other biometric modalities.

Keywords: template update, biometrics, keystroke dynamics, adaptive biometric systems, data streams

1. Introduction

Keystroke dynamics is a behavioral biometric modality that allows the recognition of individuals based on their typing rhythm on the keyboard. This biometric modality has some advantages over commonly adopted alternatives, like fingerprint or iris recognition systems [1, 2]. First, keystroke dynamics does not require an additional sensor, since a common keyboard is enough to acquire keystroke data. Second, this biometric modality can be applied in background, during other user daily tasks. These advantages may contribute to a higher acceptability of this technology.

However, as a behavioral modality, keystroke dynamics has a higher tendency to be subject to changes over time [3]. Indeed, how the user types a password evolves with time and can be different in a short timespan. The reasons are numerous and cannot always be controlled: increased practice, changes on the environment, etc. For example, users can increase the speed to write the password due to more practice. These modifications increase the intraclass variability which, consequently, can increase the ratio of authentication failure.

A strategy to reduce this performance decrease is to

*Corresponding author
Email addresses: phpisani@icmc.usp.br (Paulo Henrique Pisani), romain.giot@u-bordeaux.fr (Romain Giot), andre@icmc.usp.br (André C. P. L. F. de Carvalho), aclorena@unifesp.br (Ana Carolina Lorena)

adopt a *template update* mechanism (sometimes referred to as an *adaptive biometric system*) [4, 5]. The aim of the template update is to automatically adapt the biometric model/reference of the user to make it closer to the user current biometric data (i.e., decreasing the deviation due to template ageing). However, this update process is done without supervision (i.e., it is totally automatic), and is therefore subject to errors. This may lead to reduced predictive performance, illustrating the difficulty of this task.

There are not many studies on template update for keystroke dynamics, highlighting the need for further investigations. Before introducing the proposed approach, we should clearly specify some terms regarding the biometric samples:

- *True genuine/positive*: a biometric sample which belongs to the genuine user.
- *True impostor/negative*: a biometric sample which belongs to an impostor.
- *Classified as genuine/positive*: a biometric sample classified as genuine by the classifier. It could belong to the genuine user (the classifier returned the correct label) or it could come from an impostor (the classifier returned the wrong label).
- *Classified as impostor/negative*: a biometric sample classified as impostor by the classifier. It could belong to the genuine user (the classifier returned the wrong label) or it could come from an impostor (the classifier returned the correct label).

The majority of the papers in the area updates the user model only with biometric samples classified as genuine/positive, discarding those classified as impostors (negative). They usually employ a *positive gallery*, which is a set of biometric samples classified as genuine/positive. This paper proposes to investigate if taking into account samples classified as impostors can improve the adaptive procedure. Thus, there would be a *negative gallery* too. This new template update approach, which uses samples classified as both positive and negative for template update, is named here as *Enhanced Template Update* (ETU). The usage of negative samples in the template update process has two main motivations:

- *Reduce False Match Rate (FMR)* [1]: As some impostor samples would be available, they may help to avoid the inclusion of negative samples in the positive gallery. We propose an approach to take advantage of this concept in order to decrease the number of negative samples wrongly included in the positive gallery during adaptation. The proposed approach is named *Positive Gallery Protection* (PGP).
- *Reduce False Non-match Rate (FNMR)* [2]: This can be done by changing the classification decision. Sometimes the positive model alone (induced only from positive samples) may reject a given positive sample, but with the help of a negative model (induced from negative samples), it may be possible to verify whether the sample is closer to the positive model than the negative model. As a result, FNMR can be reduced. We propose some alternatives to change the classification decision based on this reasoning. Four different methods to change the classification decision are proposed, named as ETU 0 to 3.

It must be observed that a reduction of FNMR usually results in an increase of FMR (and vice-versa).

The contributions of this paper are:

- Proposal of a framework for template update using biometric samples classified as positive and the ones

[1] FMR measures the rate in which an impostor is wrongly accepted by the biometric system (it is an error rate, so it must be as low as possible).

[2] FNMR measures the rate in which the genuine user is incorrectly rejected by the biometric system (it is an error rate, so it must be as low as possible).

classified as negative. This may lead to further studies by the adaptive biometric community using this additional information;

- Study advantages and drawbacks of several configurations of the proposed framework;
- Performance evaluation on public main keystroke dynamics datasets, including different types of feature vectors. To the best of our knowledge, this is one of the first papers to use the passwords part of the GREYC-Web dataset [6].
- Improve the visualization of the performance of adaptive biometric methods over time.

This work does not aim at providing a new keystroke dynamics authentication algorithm neither a performance comparison of various authentication mechanisms linked to our framework. The current study is only interested in the architecture of the template update system and its application with standard authentication algorithms from keystroke dynamics literature, although our methods may be directly applied to other classification algorithms.

This paper is organized as follows: Section 2 introduces previous work on template update for keystroke dynamics; Section 3 presents the enhanced template update framework and the methods investigated in this paper; Section 4 describes the experimental methodology, including datasets, biometric data stream generation and parameters adopted in the experiments; Section 5 shows the experimental results, including a discussion on the performance over time; finally, Section 6 presents the main conclusions of this study and alternatives for future work.

2. Template Update for Keystroke Dynamics

The intra-class variation issue has been observed for various biometric technologies, like fingerprint and face recognition [4]. *Adaptive biometric systems/Template update* can deal with variations on the users characteristics by adapting the user template/model over time [5]. There are not many studies on the use of *adaptive biometric systems* in the literature; a possible reason is the lack of public datasets for these systems, which occurs for several biometric modalities [4]. These datasets have to meet some requirements, such as having several samples per user and such samples need to ideally be acquired in different sessions. Some datasets that can be used to evaluate keystroke dynamics in a template update context are described in Section 4.1.

In these adaptive biometric systems, given a set of user samples (labelled samples), a user model is initially induced and it is continuously adapted as new unlabelled samples are received during the biometric system operation. Some classical samples from literature are *self-update* (mono-modal adaptation) and *co-update* (multi-modal adaptation) [7]. The focus of this paper is on mono-modal adaptation.

Recent studies have shown that the biometric features in keystroke dynamics may change over time [8, 9], indicating the need of adaptive approaches. Some previous work on model adaptation for keystroke dynamics can be found in: [8], [9], [10] and [11]. In [10], two simple methods based on the concept of *galleries* were discussed: *growing window* and *moving window*. Later, in [8], these approaches were further investigated and a new method, known as *Double Parallel* was proposed. This method combines the concepts of growing and moving window into a new framework. According to their results, Double Parallel presented the best overall predictive performance among all tested approaches. Another paper focused on adaptation in a *free text* application [11]. In [9], the authors proposed *Usage Control R*, which adapts the user model based on the usage of detectors.

Next sections briefly describe some of these adaptive methods. All of the adaptive methods described here follow the general flow presented in Figure 1. In this flow, user model/template can be updated every time a query

sample is classified as positive/genuine. It must be observed that this may result in errors, as the classifier may not correctly classify all samples.

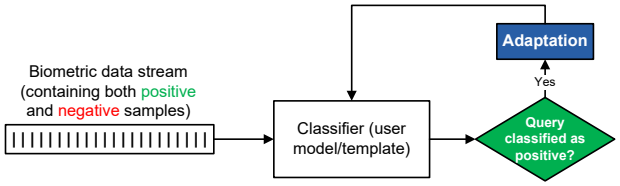


Figure 1: Basic template update flow of the studied algorithms.

2.1. Model-based adaptive methods

Model-based adaptive methods involve retraining the classification algorithm to adapt the user model over time. In [8], a method which reached good overall performance was *Double Parallel* and is part of our experiments.

Double Parallel keeps two user models in memory: one is generated using growing window and another one uses sliding window. In the growing window model, all training samples are stored in memory and they are used to induce it. Afterwards, in the matching phase, any sample recognized as positive (genuine user) whose score is higher than an update threshold is added to the set of samples and the user model is re-trained. The sliding window model follows a similar procedure, but it also removes the oldest training sample before retraining the algorithm. As *Double Parallel* keeps two user models, the query sample is presented to both models, producing two score outputs. For this study, the algorithm was implemented in the version which computes the average between both scores and uses it to perform classification.

Although *Double Parallel* (DB) presented good predictive performance in the experiments reported in the literature, its memory usage can grow without limit over time. It is due to the use of a model induced by the growing window method, which does not remove samples from memory. Later, in [13], the authors modified Double Parallel when used with M2005 in order to make the algorithm incremental. This new version, named IDB, solves the memory usage problem by updating the growing model incrementally, updating mean and standard deviation. As median could not be updated incrementally, mean was used instead. Both DB and IDB adapt the user model using only use samples classified as positive/genuine.

2.2. Detector-based adaptive methods

Similarly to model-based, detector-based adaptive methods also only update the detectors when a query sample is classified as positive/genuine. Two simple detector-based adaptation methods are: *Growing* and *Sliding*. They were implemented based on [8] and [10], which used similar ideas. Their performance were evaluated in [9]. In the *Growing* version, each sample classified as being from the genuine user (positive) is included as a new detector. The *Sliding* version works in the same way. However, it also discards the oldest detector when a new detector is added. This makes the amount of detectors constant and is, therefore, more efficient than *Growing* regarding memory usage, which only grows the detector set. Due to this problem with *Growing* over time and considering that *Sliding* performed better than *Growing* in previous studies [9], only *Sliding* is used here.

Another adaptive algorithm is *Usage Control R* [9]. This algorithm assesses which detectors are more used in order to decide whether to keep them in the detector set. The storage of samples in memory and their replacement according to their usage was also discussed in the context of biometrics in a technical report [15], although their approach is different from *Usage Control*. For each detector, two new attributes are assigned in *Usage Control*:

- *Usage count*: increases every time the detector matches a query sample.
- *Recent usage*: decreases when another detector matches a query sample. If the detector matches the query sample, it returns to a maximum value (here we adopted 10, the same value adopted in [9]). When

the detector is firstly generated, it also assumes the maximum value.

In *Usage Control R*, when a new sample is presented, if a detector matches it, the additional attributes are updated (i.e., only when the sample is classified as positive). Detectors are checked from the newest to the oldest one. The first one to match the query sample is considered as "used". All detectors with *Recent usage* = 0 are ordered by *Usage count*. The detector with the lowest *Usage count* is removed and a new detector is added to the set using the matched sample. The effect of these additional attributes in *Usage Control R* is the removal of detectors with low usage without removing new detectors instantly (as their *Usage count* is zero when they are created).

Later, *Usage Control S* was proposed in [16]. This algorithm only updates the detector set if at least two detectors are able to match the query sample. By doing so, it is assumed that a sample matched by two or more detectors has a higher level of confidence that it is a true positive. Similarly, samples matched by only one sample are assumed to have low level of confidence and, therefore, are not used as a new detector. In addition, *Usage Control S* consider all detectors that are able to match the query sample as "used" (this is different from *Usage Control R*, which only considers the first detector to match as "used"). This avoids the removal of detectors that could represent well the current user behaviour, although they were not the first to match.

In both *Usage Control R* and *S*, when there is no detector with *Recent usage* equals to zero, no adaptation occurs and the recognized sample is discarded. However, this could lead the algorithm to lose key information for adaptation when faced to small changes. To overcome this problem, *Usage Control 2* adds all matched samples as detectors, regardless of the *Recent usage* values. However, this could result in an endless increase in the set of detectors (when new detectors are included and no detector has *Recent usage* = 0). This is related to the behaviour of the *Growing* approach. In order to avoid this problem, whenever a new sample is recognized as positive, the *Usage Control 2* algorithm, instead of just removing a single detector (the one with least *Usage count*), it removes all detectors with *Recent usage* = 0. As a result, the set of detectors in *Usage Control 2* can increase (when no detector has *Recent usage* = 0) or decrease (when more than one detector has *Recent usage* = 0). Consequently, the number of detectors is not constant, different from *Usage Control R* or *S*.

3. Proposal of Enhanced Template Update

This section presents the *Enhanced Template Update* framework. The main flow is shown in Figure 2. Firstly, at enrollment/training phase, it stores all genuine training biometric samples and induces the positive classifier. Afterwards, during the recognition phase, query samples will be received. Initially, if they are classified as positive, they are added to the positive gallery (the older sample is also removed from the gallery) and the positive classifier is updated. Otherwise, if the sample is classified as negative, it is added to the negative gallery. Until this stage, the behaviour of the template update process is similar to a standard *Self-update* procedure, retraining the positive algorithm based on the updated gallery (although the standard *Self-update* does not store negative samples).

After a minimum amount of samples classified as negative is obtained, the negative procedure is enabled. The negative classifier is then induced and updated similarly to the way the positive one is. If the target amount of samples classified as negative has been obtained, the oldest sample is removed from the negative gallery every time a new negative sample is added. When the negative procedure is enabled, the classification of the new query samples is based on both the positive and the negative models. Note that our proposal is different from [17], which only updated the negative gallery (named as impostor database) to retrain novelty detectors. Enhanced Template Update

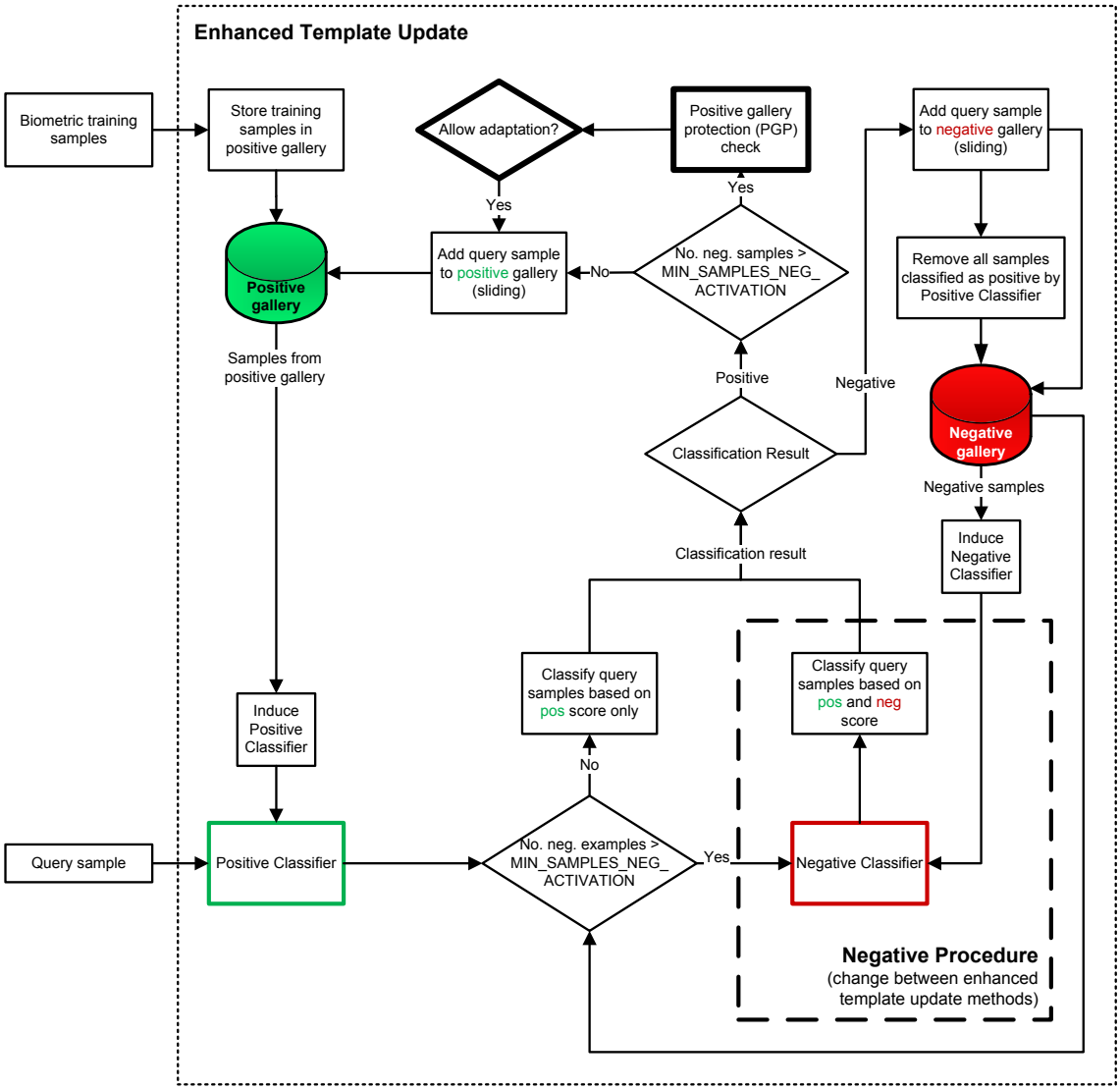


Figure 2: Enhanced Template Update Framework. It makes use of two galleries: one for samples classified as positive and another for the samples classified as negative. There are four ETU methods: ETU 0 to 3. They basically change the Negative Procedure, highlighted in the figure.

(ETU) makes use of two models/galleries to support classification and adaptation.

The way the classification decision is taken will vary among the four ETU methods: ETU 0 to 3. These four ETU methods can work with either model or detector based techniques, such as M2005 [12] or *Self-Detector* [14] (using cosine distance). These are two static algorithms previously used under adaptive approaches for keystroke dynamics. Hence, we can directly compare the performance of ETU methods to current adaptive approaches using the same reference static algorithms. However, the methods proposed here can also be applied to other classification algorithms, as long as they output a similarity score (the single exception is ETU 2).

3.1. Positive Gallery Protection Check

Enhanced Update Template framework includes an optional *Positive Gallery Protection (PGP)* check. When activated, it attempts to avoid the inclusion of negative samples in the positive gallery. Note that the negative gallery must reach a minimum amount of samples, as PGP uses them to support its decision.

It works in the following way. First, it clusters all samples in the negative gallery using *K-Means++* [18]. *KMeans++* is a *K-Means* [19] variant which is less prone to poor random initialization of the centers. We applied a simple algorithm just to illustrate that it may improve the performance under the given scenario. Other clustering algorithms may be investigated in the future. The k parameter is tuned according to the method described in Section 3.5. Afterwards, it computes the center of all obtained negative clusters (*negative centers*). In addition, the whole positive gallery is considered a single positive cluster and a *positive center* is also obtained.

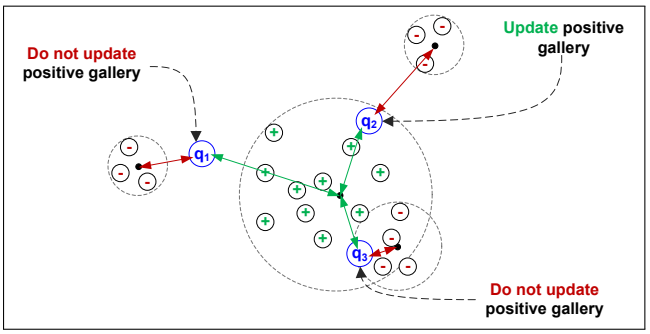


Figure 3: Enhanced Template Update - Positive Gallery Protection (PGP). Each positive circle (+) represents a sample in the positive gallery and each negative circle (−) represents a sample in the negative gallery. (q) is the newly classified sample, the query (there are three queries in the figure to represent three situations). Note that if the closest center is a negative one, positive gallery is not updated.

After obtaining the centers, PGP looks for the cluster center which is closest to the query sample classified as positive. If the closest cluster is the positive cluster, the query sample can be added to the positive gallery, and, therefore, the positive model is adapted. Otherwise, if the closest cluster center is a negative one, the positive gallery is not updated. Some hypothetical situations are shown in Figure 3 to illustrate the PGP check. Query 1 is easy to identify as it is outside of the positive cluster. However, queries 2 and 3 are inside the positive cluster, so they are likely to be true positive samples. As there is an overlapping negative cluster closer to query 3, this sample is not used to update the positive gallery.

3.2. ETU 0: Simple Comparison of Scores (Model and Detector)

This was the first method evaluated. As it is the simplest one, it is called ETU 0. It adopts a simple rule: classify the query sample as positive if the positive score is higher than the negative score (Figure 4). Note that this method is applicable to both Model and Detector-based methods, as it is possible to obtain a score from these two algorithms. For Detector-based, the *score* is the correlation between the query sample and the closest detector to the query sample.

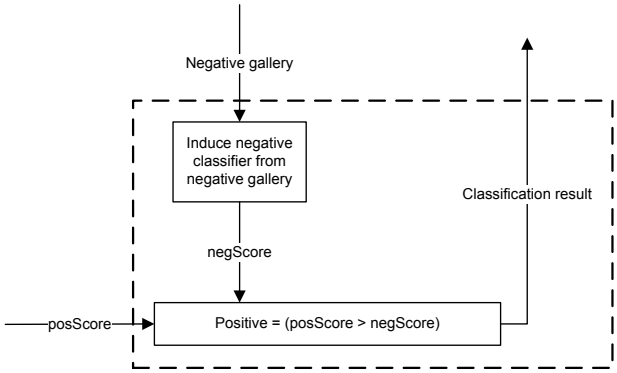


Figure 4: Enhanced Template Update - Method 0.

However, this method may not be suitable to be used with model-based algorithms, particularly M2005. The negative gallery has samples from several different users. Thus, the standard deviation on the negative gallery is probably high and it can potentially result in a misleading high score for several samples. Consequently, this simple rule may result in high FNMR. A solution for this issue is presented in Section 3.5, which discusses ETU 3.

3.3. ETU 1: Simple Comparison of Scores (Detector)

It is an incremental modification over ETU 0 for Detector-based methods (Figure 5). Now, the sample is classified as positive if the positive score is higher than the negative score and if the difference between them is higher than twice the self-radius. As a result, the classification becomes more rigorous, which may contribute to decrease false match in Detector-based algorithms.

In some situations, an impostor query sample can be very far from both the positive and the negative model, although it may be closer to the positive model. In this case the sample should be rejected, but it would be accepted in ETU 0 as $positiveScore > negativeScore$ (even though both are low score values in this hypothetical example). To avoid this issue, we propose to check if the difference between the scores is large enough. This method was designed to deal with a possible problem of ETU 0 when applied to Detector-based methods. However, it can be applied to other algorithms using the similarity score instead of the self-radius.

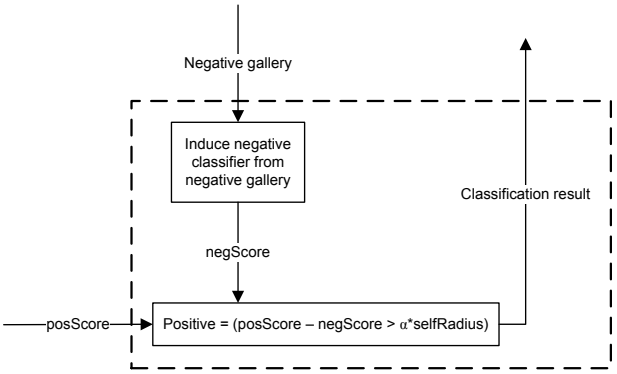


Figure 5: Enhanced Template Update - Method 1.

3.4. ETU 2: k-NN like (Detector)

Detector-based algorithms can be understood a instance-based algorithms. Thus, if we group positive and negative detectors (from positive and negative galleries), as shown in Figure 6, the k-Nearest Neighbour algorithm can be used. If most of the k closest detectors are positive, then the query sample is classified as positive, otherwise, as negative (note that if k is even, a draw can happen. In this case, we opted to classify the sample as negative in this case since it is likely to be a impostor attempt).

3.5. ETU 3: Clustering Negative Samples (Model)

As discussed earlier, the negative gallery may have high standard deviation, implying in high negative scores for model-based algorithms, particularly for M2005. ETU 3

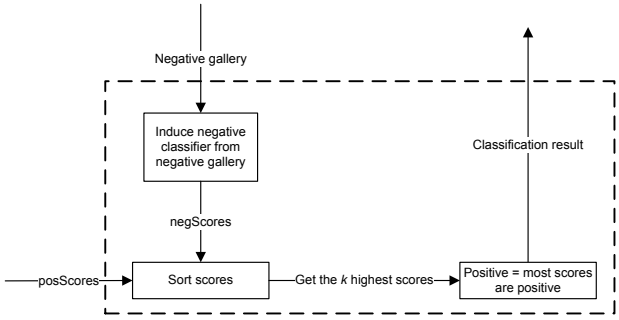


Figure 6: Enhanced Template Update - Method 2.

deals with this problem by applying clustering to the negative gallery (Figure 7). For each cluster, a negative model is generated using M2005. To classify a query sample, it is tested against each negative model. The negative score is the average of the scores output from all negative models. This method was designed based on a possible issue of M2005 in ETU 0, so this algorithm is only applied to Model-based in this work.

The use of clustering reduces the standard deviation of the sample sets. *KMeans++* [18] algorithm was used.

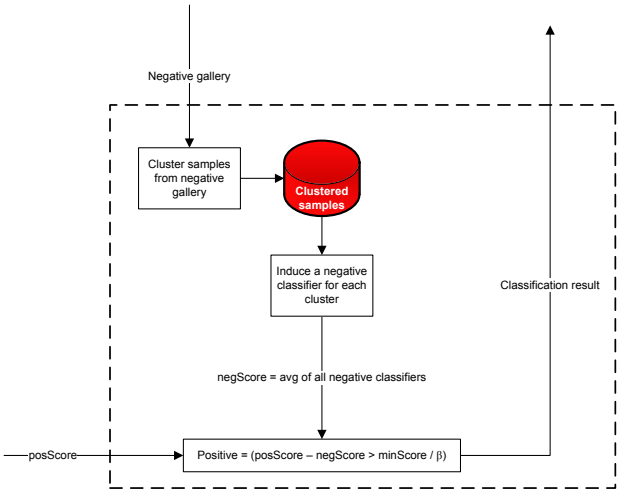


Figure 7: Enhanced Template Update - Method 3.

A method based on OMRk (*Ordered multiple runs of k-means*) [20] was used tune the k parameter of *KMeans++*. This method executes the clustering algorithm for several values of k and selects the value using a clustering validity index (k ranged from 3 to the negative gallery size / 2). The validity index used here is based on the maxi-

mum standard deviation among all obtained clusters. If this maximum standard deviation is smaller than the *target* value, the current k value is returned. The *target* value is the standard deviation in the positive gallery, which is considered as a single cluster in this case. We adopted this strategy in order to reduce standard deviation in the clusters and possibly avoid the ETU 0 problem when applied to model-based algorithms.

4. Experimental Methodology

This section presents the experimental methodology adopted to evaluate the proposed ETU methods.

4.1. Datasets

In order to facilitate the reproducibility of the experiments, the following public datasets were used. Datasets for the study of template update need to meet some requirements, such as containing several samples per user and these samples should ideally be acquired at different sessions.

- **CMU** [21]: 51 users typed the password ".tie5Roanl" plus the *Enter* key 400 times in eight sessions. Considering all users, a total of 20,400 samples are available in this dataset.
- **GREYC-Web** [6]: 118 users contributed to this dataset, some of them for more than 1 year. The updated version, available in the authors website, was used here. This dataset has data for logins and passwords. Hence this dataset can be separated into two datasets:
  - **Logins**: for the transcription of the login ("laboratoire greyc"), we considered the 35 users with at least 100 valid samples. This results in more than 7,000 samples.
  - **Passwords**: for the transcription of the passwords ("SÉSAME"), we considered the 29 users with at least 100 valid samples. This results in more than 5,500 samples. To the best of our knowledge, this is the first study to use the passwords part of this dataset.

Another important public dataset for keystroke dynamics is GREYC [22]. However, this dataset has an average of 67.5 samples per user (only one user has more than 100 samples and we consider that this is not enough for our study). To the best of our knowledge, the datasets mentioned here are the only ones publicly available that have enough data for a study of template update over time.

From these datasets, the feature vectors are extracted. *Self-Detector* uses two feature vectors here, both using order-based techniques: *flight time* type 1 using rank transformation [23] and nGdv [24]. *Flight time* type 1 [25] computes the time difference between the instants when a *key* is released and the next *key* is pressed. According to [26], this is one of the most used features in previous keystroke dynamics studies. Over these data, rank transformation is applied. Regarding nGdv, it considers the time difference between the instants that consecutive keys are pressed (*n-graphs*). Our experiments considered digraphs, which measure n-graphs for each two consecutive keys, the same that the original nGdv paper did. The time differences are then ranked and transformed using inequalities. Additional details of nGdv are presented in [24]. M2005, however, uses only raw data, since our preliminary experiments have shown that raw data results in higher accuracy than order based data for this algorithm. In order to allow a direct comparison with previous work using M2005, *flight time* type 1 raw data is used [13].

4.2. Evaluation Methodology

The user model/template is induced only by the first genuine samples (*training* phase) for each positive user. Afterwards, in the *test* phase, a biometric data stream is generated as described in the next section. The generated data stream is then presented to the classifier, sample by

sample, which will perform classification and adapt the user model.

The samples in the data stream do not have a class label, thus, the classifier does not know their true label. Several studies in the area of data stream mining assume that the true label is provided to the classifier after the classification. However, in this study, the true label is never provided to the classifier. This is done on purpose, to perform an experiment closer to a biometrics practical scenario. It is also important to highlight that, when generating a data stream for a given positive user, samples already used for *training* are not part of the data stream for that user.

The reported results shown ahead in this paper are the average values of the performance measures considering all users, since the test is performed per user. Furthermore, due to the stochastic nature of the data stream generation (true positive and true negative samples are interleaved randomly), all experiments are repeated 30 times. Next section describes how the biometric data stream is generated.

4.3. Biometric Data Stream Generation with User-crossvalidation

In this paper, the biometric data streams generated are based on the *user cross-validation* methodology presented in [13]. This methodology divides the users into N groups of similar size (N assumed the value 5 in this paper, as in [13]). Based on these N groups, N test scenarios are evaluated. For each scenario, the users in the $(N-1)$ groups form the *positive set* and the remaining group form the *negative only set*. Next section describes the meaning of each set and how the data stream is generated.

The *positive set* has users that will be tested as genuine users, so a biometric data stream will be generated for each of them. These users can be understood as the employees from a company that are enrolled in the biometric system. During the experiments, some of these genuine users may be attacked by other genuine users (e.g. some employees may want to attack other employees accounts). By doing this, we are able to simulate *internal attacks* (*insiders*).

The other set is the *negative only*, which has impostor only users. These impostors are used to perform attack simulation from unknown users. Following the same example of the company, these negative only users are not enrolled in the biometric system. They would be people that do not work in the company and want to attack the system. As a result, we can also simulate *external attacks*.

4.3.1. Biometric Data Stream

As previously mentioned, a data stream is generated for each user in the *positive set*. The generated data stream is formed by all test samples from the genuine user randomly interleaved with samples from impostors. The biometric stream has 70% of genuine samples and 30% of impostor samples, as previously considered in keystroke dynamics studies [8, 9, 13]. Among the 30% negative samples, there is a 50% chance of getting a impostor/negative sample from the *negative only set* (external attack) and a 50% chance of getting a negative sample from the *positive set* (internal attack). For all users (including impostors), the order in which the samples appear in the dataset is maintained. This is a key aspect, as it allows to verify possible concept drift/change in the way the user types on the keyboard over time.

4.4. Authentication Algorithms

This paper uses one-class classification algorithms already used in previous work dealing with template update for keystroke dynamics [8, 9, 13]. These algorithms are: M2005 [12] for Model-based and *Self-Detector* for Detector-based [14]. These algorithms were chosen because they have been used in previous work dealing with template update for keystroke dynamics. By using them, it is easier to compare the obtained results to other papers in the area of template update. They are both described in the next sections.

4.4.1. M2005 Algorithm

This algorithm, named M2005 in this paper, was proposed by [12] for keystroke dynamics recognition. It extracts statistical values from the training samples for each feature (mean, median and standard deviation), which are used to represent the user model. Afterwards, in a test phase, M2005 verifies each feature of the given sample to check if it meets conditions shown in (1) and (2), in which d_i is the value of the feature i in the given sample and $mean_i$, $median_i$ and std_i are the mean, median and standard deviation, respectively, of the feature i from the training samples.

$$min(mean_i; median_i) * (0.95 - std_i/mean_i) <= d_i \quad (1)$$

$$d_i <= max(mean_i; median_i) * (1.05 + std_i/mean_i) \quad (2)$$

For each feature i which satisfies conditions (1) and (2), the algorithm computes a sum according to the following rules:

- if d_i is the first feature, 1.0 is added to the sum;
- if $d_{(i-1)}$ does not meet (1) and (2), 1.0 is added to the sum;
- if $d_{(i-1)}$ also meets (1) and (2), 1.5 is added to the sum.

After verifying all features of the sample, the algorithm computes a score using Equation (3), in which max_sum is defined as $1.0 + 1.5 * (feature\_count - 1.0)$.

$$Score = sum/max\_sum \quad (3)$$

For the classification of a new sample, if the computed score is higher than a given threshold, the sample is classified as positive (genuine user) and, otherwise, as negative (impostor).

4.4.2. Self-Detector Algorithm

Another algorithm used in the context of template update for keystroke dynamics is *Self-Detector*, which is an immune algorithm of the positive selection class. The standard *Self-Detector* [14] uses training samples from the genuine user as detectors and assigns a radius to each of them. Whenever a query sample is presented to the classifier, all detectors are tested against it. If any detector matches the query sample, it is classified as *self* (genuine user), otherwise, as *non-self* (impostor). In this study, a detector matches a sample if the distance between its center and the sample is smaller than its radius. The original version of this algorithm uses a ROC analysis to define the radius. A different approach is used here, as described in Section 4.5. An overview of the behaviour of the algorithm under an adaptive context is presented in Figure 8.

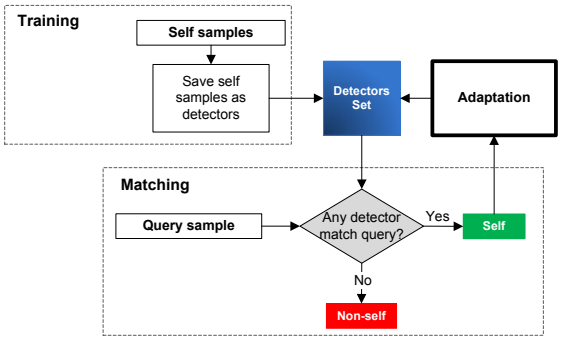


Figure 8: Adaptive Positive Selection Algorithm (figure from [9]). Detectors may be updated when the query sample is classified as *Self*.

4.5. Parameters

The current study has adopted the same parameter values from [9] and [13] for the classification algorithms. GREYC-Web Passwords has not been used in those studies, so we applied the same parameter tuning method from [13] using the current *user crossvalidation* approach. *Self-Detector* algorithms used cosine distance, so the clustering algorithm adopted the same distance measure when applied to *Self-Detectors*. We employed the *K-Means++* implementation from *Apache Commons* [27]. A summary of the parameter values are shown in Tables 1 and 3.

Regarding the size of the positive and negative galleries, we recommend that both should be of the same size. In

our experiments, 40 samples were used for training and this defines that the positive gallery has 40 samples. Consequently, the minimum size of the negative gallery to activate ETU methods should also be 40. However, several GREYC-Web users have far less samples than the users in CMU. As a result, using 40 as the minimum for GREYC-Web would imply that the negative procedure would not be activated for some users. In order to avoid this situation, the minimum amount of samples for activating the negative procedure is 20 in GREYC-Web (the maximum size was kept at 40). For CMU, we maintained 40 as the minimum since all users have 400 samples, which can enable ETU. Decreasing the minimum size lead to a imbalance between the positive and negative galleries (negative gallery would be used with 20 samples while positive gallery contains 40 samples), hence it can imply in reduced predictive performance.

Table 1: Parameter values for the classification algorithms.

	Self-Detector (Detector-based)	**M2005** (Model-based)
Dataset	*Self-radius*	*Threshold*
CMU	0.02	0.7
GREYC-Web Logins	0.04	0.6
GREYC-Web Passwords	0.01	0.6

Table 2: ETU - Negative gallery.

Dataset	**Minimum size for activating negative procedure**	**Maximum size**
CMU	40	40
GREYC-Web Logins	20	40
GREYC-Web Passwords	20	40

Table 3: Parameter values specific of ETU methods.

Method	Parameters
ETU 0	-
ETU 1	$\alpha = 2$
ETU 2	$k = 2$
ETU 3	$\beta = 2$

4.6. Performance Measures

In the experimental results, the following performance measures were adopted:

- False Match Rate (FMR);
- False Non-Match Rate (FNMR);
- Balanced Accuracy: combines FMR and FNMR in a single rate and is defined here as $1.0 - (FMR + FNMR)/2.0$.

Throughout the paper, we present results for these measures globally (average results over the whole biometric data stream) and over time.

4.7. Evaluation Over Time

In order to evaluate the performance over time, we used an improved version of the methodology adopted in a previous paper [13]. The measures FNMR and FMR are extracted using a window of size 50 in steps of 10 samples. The measured values are plotted in the graph. The average performance over all positive users of the first group division of *user cross-validation* is reported. This procedure allows to see the rates measured through the biometric data stream. However, if each positive user has a different number of samples in the data stream, the graph is limited to the shortest stream. Otherwise, the later parts of the graph would be the average of a decreased number of users. It is how the previous version of this evaluation methodology worked.

The new version of this graph, introduced in this work, does not stop at the shortest stream, but, instead, presents average values for all available users. Since, in the later parts of the stream, a reduced number of users may be part of the results, the graph also shows the interval based on the *standard error of the mean* (shaded area), as described in Equation 4 (CI_i : confidence interval), which makes use of SE calculated in Equation 5. In Equation 5, std_i is the standard deviation among the measures at

window i and $users_i$ is the number of users with available data at window i. This does not restrict the graph to the shortest data stream and provides additional results to support the experimental analysis. The new graphs are shown on Section 5.2.

$$CI_i = mean(measure)_i \pm 1.96 * SE_i \quad (4)$$

$$SE_i = std_i/\sqrt{users_i} \quad (5)$$

5. Experimental Results

This section presents the results obtained in the experiments. It starts with the global results. Next, a predictive performance over time is presented and discussed.

5.1. Global Results

Initially, overall results for both datasets are shown in Tables 4, 5 and 6. These tables show the results for FMR, FNMR and *Balanced Accuracy*. Both FMR and FNMR are error rates, so they should be as low as possible. PGP was applied to all adaptive algorithms which use a single positive gallery. That is the case for all ETU methods and *Sliding*. In these Tables, the detector-based (*Self-Detector*) and the model-based (M2005) algorithms are grouped. Detector-based algorithms used both flight time and nGdv, while model-based only used flight time. As described in Section 4.1, M2005 performs better using raw data while Self-Detector benefits from the use of order-based feature vectors. The baseline in the tables is the static algorithm of each group. Hence, for instance, ETU 2 should perform better than *Self-Detector* (no adaptation). Similarly, ETU 3 should perform better than M2005 (no adaptation).

Some tendencies can be observed in the global results. In general, all ETU methods without PGP presented a better predictive performance than the static algorithm, with the exception of ETU 0 and 1, which had a lower accuracy performance for the GREYC-Web dataset. It is interesting that even for a small password ("SÉSAME"), the accuracy increased when an adaptive algorithm was used. Therefore, it indicates that the typing behaviour affects the predictive performance even for short sentences. When compared the predictive performance of *Self-Detector* against M2005, the static *Self-Detector* (flight time) obtained higher balanced accuracy than static M2005 in all datasets. When their adaptive algorithms are applied, M2005-based tend to obtain a higher performance difference between the static and the adaptive algorithm. In CMU and GREYC-Web Passwords dataset, for example, this higher performance gain resulted in better accuracy for adaptive M2005 when compared to adaptive *Self-Detector*.

Adaptive approaches managed to improve FNMR in most cases. However, this was not true for some adaptive algorithms in GREYC-Web Passwords. The reason can be the high FMR in the static algorithm, which may have contributed to the inclusion of many impostor samples in the positive model. As a result, the positive model has become far from the genuine user.

ETU 0 was applied to both *Self-Detector* and M2005, but FMR and FNMR behaviours were the opposite. M2005 managed to decrease FMR at the cost of the highest FNMR. This may be explained by characteristics of the M2005 algorithm, which increases the rate of acceptance as the standard deviation increases. It is expected that the standard deviation increases in ETU 0, as it stores negative samples from several different users in the negative gallery. As a result, the negative model has a tendency to return higher scores, contributing to the rejection of more users.

Self-Detector may be affected by another issue in ETU 0. In some cases, a true negative query may be very far from both the positive and the negative models, although it may be closer to the positive model (remember that the score in the *Self-Detector* is the correlation between the query sample and the closest detector). In this case, the sample should be rejected, but it is accepted in ETU 0 as $positiveScore > negativeScore$ (even though both are

Table 4: Global results for the CMU dataset (best results in bold and standard deviation between parenthesis). Both baseline non-adaptive algorithms are highlighted in bold and their respective adaptive methods are shown below. The methods proposed in this paper are denoted as ETU. For detector-based algorithms, results for both flight time and nGdv are reported.

CMU Dataset								
			Without PGP			With PGP		
		Algorithm	**FMR**	**FNMR**	**Acc (balanc.)**	**FMR**	**FNMR**	**Acc (balanc.)**
Flight time		***Self-Detector (No adaptation)***	0.287 (0.023)	0.410 (0.016)	0.651 (0.009)			
		Self-Detector (Sliding)	0.291 (0.031)	**0.211** (0.013)	0.749 (0.016)	0.250 (0.021)	0.251 (0.015)	0.750 (0.011)
		Self-Detector (Usage Control 2)	**0.143** (0.012)	0.323 (0.014)	**0.767** (0.009)			
		Self-Detector (Usage Control R)	0.311 (0.030)	0.220 (0.013)	0.735 (0.015)			
		Self-Detector (Usage Control S)	0.213 (0.014)	0.275 (0.012)	0.756 (0.008)			
	Proposals	*Self-Detector (ETU 0)*	0.538 (0.016)	**0.102** (0.007)	0.680 (0.009)	0.573 (0.018)	**0.088** (0.009)	0.670 (0.009)
	Proposals	*Self-Detector (ETU 1)*	**0.251** (0.015)	0.203 (0.017)	**0.773** (0.013)	**0.271** (0.012)	0.201 (0.015)	**0.764** (0.010)
	Proposals	*Self-Detector (ETU 2)*	0.285 (0.019)	0.207 (0.013)	0.754 (0.013)	0.268 (0.016)	0.224 (0.014)	0.754 (0.011)
nGdv		***Self-Detector (No adaptation)***	0.281 (0.018)	0.457 (0.012)	0.631 (0.008)			
		Self-Detector (Sliding)	0.327 (0.030)	**0.232** (0.009)	0.721 (0.016)	0.290 (0.022)	0.252 (0.011)	0.729 (0.012)
		Self-Detector (Usage Control 2)	**0.157** (0.014)	0.355 (0.010)	**0.744** (0.010)			
		Self-Detector (Usage Control R)	0.339 (0.029)	0.237 (0.009)	0.712 (0.015)			
		Self-Detector (Usage Control S)	0.247 (0.018)	0.283 (0.010)	0.735 (0.011)			
	Proposals	*Self-Detector (ETU 0)*	0.581 (0.017)	**0.110** (0.007)	0.654 (0.008)	0.602 (0.022)	**0.094** (0.008)	0.652 (0.009)
	Proposals	*Self-Detector (ETU 1)*	**0.260** (0.028)	0.378 (0.014)	0.681 (0.011)	**0.260** (0.028)	0.378 (0.014)	0.681 (0.011)
	Proposals	*Self-Detector (ETU 2)*	0.319 (0.019)	0.223 (0.009)	**0.729** (0.012)	0.304 (0.017)	0.231 (0.010)	**0.732** (0.011)
		M2005 (No adaptation)	0.273 (0.028)	0.451 (0.019)	0.638 (0.013)			
		M2005 (DB)	0.129 (0.014)	0.373 (0.014)	0.749 (0.010)			
		M2005 (IDB)	**0.122** (0.011)	**0.306** (0.008)	**0.786** (0.006)			
	Proposals	*M2005 (ETU 0)*	**0.064** (0.008)	0.623 (0.019)	0.656 (0.009)	**0.064** (0.009)	0.669 (0.015)	0.633 (0.008)
	Proposals	*M2005 (ETU 3)*	0.244 (0.016)	**0.143** (0.006)	**0.807** (0.009)	0.175 (0.017)	**0.243** (0.011)	**0.791** (0.010)

low score values in this hypothetical example). To make things worse, when this occurs, the sample is added to the positive gallery, contributing to increase FMR.

An important result for balanced accuracy was obtained by ETU 3, which, apart from improving FNMR, obtained the highest accuracy in CMU. ETU 3 was also the best ETU method in GREYC-Web Passwords. Conversely, for GREYC-Web Logins, *Double Parallel* methods were better than ETU 3. This indicates that ETU 3 (which uses clustering) managed to improve the performance over ETU 0.

Regarding the positive gallery protection check, according to the results, it reduced the FMR in several cases. The main exception is for ETU 0. However, it may be a result of the high number of classification errors obtained by ETU 0. ETU framework is based on sliding gallery management, so we also applied PGP to the standard *Sliding Self-Detector*. Even for this algorithm, FMR managed to be reduced on both datasets. Nevertheless, this implied in a slightly decrease of the balanced accuracy. It suggests that some positive samples may have been wrongly rejected by PGP. The main benefit was observed for ETU 3 and *Self-Detector* Sliding, which reduced FMR up to approximately 6% depending on the dataset.

When *Self-Detector* is referenced here, the conclusions for flight time and nGdv are almost always the same. This is an interesting result, since it illustrates that even using different feature vectors, behaviour change occurs and the adaptive methods have similar tendencies of improvement. Although the adaptive methods resulted in similar improvements on both feature vectors, it must be observed that the overall performance of flight time is higher than that of nGdv in our scenario. It does not mean that nGdv is not an appropriate solution, as it has proven a good feature vector in previous work using free text [24]. However, some important remarks must be made. First, the

Table 5: Global results for the GREYC-Web Logins datasets (best results in bold and standard deviation between parenthesis). Both baseline non-adaptive algorithms are highlighted in bold and their respective adaptive methods are shown below. The methods proposed in this paper are denoted as ETU. For detector-based algorithms, results for both flight time and nGdv are reported.

GREYC-Web (logins) Dataset								
			Without PGP			With PGP		
		Algorithm	**FMR**	**FNMR**	**Acc (balanc.)**	**FMR**	**FNMR**	**Acc (balanc.)**
Flight time		***Self-Detector (No adaptation)***	0.066 (0.008)	0.141 (0.005)	0.896 (0.005)			
		Self-Detector (Sliding)	0.074 (0.011)	**0.085** (0.004)	0.920 (0.007)	0.067 (0.010)	0.106 (0.008)	0.913 (0.008)
		Self-Detector (Usage Control 2)	**0.035** (0.007)	0.148 (0.010)	0.908 (0.007)			
		Self-Detector (Usage Control R)	0.069 (0.009)	0.086 (0.004)	**0.922** (0.006)			
		Self-Detector (Usage Control S)	0.053 (0.007)	0.123 (0.005)	0.912 (0.005)			
	Proposals	*Self-Detector (ETU 0)*	0.353 (0.030)	**0.034** (0.011)	0.807 (0.017)	0.355 (0.032)	**0.042** (0.010)	0.802 (0.020)
	Proposals	*Self-Detector (ETU 1)*	**0.065** (0.013)	0.146 (0.020)	0.894 (0.011)	**0.064** (0.013)	0.151 (0.019)	0.893 (0.010)
	Proposals	*Self-Detector (ETU 2)*	0.103 (0.014)	0.071 (0.009)	**0.913** (0.010)	0.099 (0.014)	0.078 (0.011)	**0.911** (0.011)
nGdv		***Self-Detector (No adaptation)***	0.158 (0.010)	0.104 (0.007)	0.869 (0.007)			
		Self-Detector (Sliding)	0.190 (0.022)	0.058 (0.004)	0.876 (0.013)	0.171 (0.017)	0.059 (0.004)	0.885 (0.010)
		Self-Detector (Usage Control 2)	**0.094** (0.011)	0.097 (0.007)	**0.905** (0.008)			
		Self-Detector (Usage Control R)	0.175 (0.018)	**0.057** (0.004)	0.884 (0.011)			
		Self-Detector (Usage Control S)	0.144 (0.010)	0.066 (0.004)	0.895 (0.006)			
	Proposals	*Self-Detector (ETU 0)*	0.381 (0.024)	**0.023** (0.002)	0.798 (0.012)	0.379 (0.026)	**0.022** (0.002)	0.799 (0.013)
	Proposals	*Self-Detector (ETU 1)*	**0.089** (0.016)	0.591 (0.033)	0.660 (0.012)	**0.089** (0.016)	0.591 (0.033)	0.660 (0.012)
	Proposals	*Self-Detector (ETU 2)*	0.186 (0.015)	0.042 (0.004)	**0.886** (0.008)	0.183 (0.015)	0.044 (0.006)	**0.887** (0.008)
		M2005 (No adaptation)	0.096 (0.013)	0.245 (0.016)	0.829 (0.008)			
		M2005 (DB)	**0.083** (0.012)	0.179 (0.012)	0.869 (0.008)			
		M2005 (IDB)	0.095 (0.015)	**0.131** (0.011)	**0.887** (0.008)			
	Proposals	*M2005 (ETU 0)*	**0.073** (0.011)	0.427 (0.046)	0.750 (0.021)	**0.067** (0.010)	0.505 (0.037)	0.714 (0.018)
	Proposals	*M2005 (ETU 3)*	0.163 (0.017)	**0.118** (0.018)	**0.860** (0.013)	0.120 (0.016)	**0.151** (0.018)	**0.865** (0.013)

referred work for nGdv has not considered a template update scenario, which requires a different methodology. As described in Section 4, our paper adopts an evaluation methodology which uses only the first samples for training/enrollment and applies the remaining ones for test, in the form of a biometric data stream. This is done to check possible typing rhythm change over time. Second, the paper which proposed nGdv used a dataset for free text, which contains samples with 700 to 900 characters. This is a completely different case than that of our datasets for template update, which are composed by short fixed expressions. For instance, the largest expression is from GREYC-Web Logins, which has 17 characters. Due to these reasons, the remainder of the paper only discusses the results for flight time.

To evaluate the statistical significance of the results, a statistical test (*Wilcoxon Signed Rank Test*) [28] was applied with $\alpha = 0.05$. *Holm's* correction [29] was used as we are doing multiple comparisons. The results are shown in Table 7.

This table shows the ETU proposals (without PGP) compared to two baselines: one static and one adaptive. The static baseline for *Self-Detector* based algorithms is the standard *Self-Detector* (no adaptation) and, for the M2005 based algorithms, the baseline is the standard M2005 (no adaptation). The adaptive baseline is the adaptive algorithm which obtained the best balanced accuracy (considering an average on all tested datasets). For *Self-Detector*, it is the *Sliding* algorithm. For M2005, is the IDB (incremental version of *Double Parallel*). The results of the statistical test are shown per measure. A positive sign (+) means the performance of ETU is statistically better than the baseline, while a point sign (.) shows that there is no statistical evidence that ETU performs better than the baseline. The statistical test results show the ETU proposals mainly improve FNMR, in which

Table 6: Global results for the GREYC-Web Passwords (best results in bold and standard deviation between parenthesis). Both baseline non-adaptive algorithms are highlighted in bold and their respective adaptive methods are shown below. The methods proposed in this paper are denoted as ETU. For detector-based algorithms, results for both flight time and nGdv are reported.

GREYC-Web (passwords) Dataset								
			Without PGP			With PGP		
		Algorithm	**FMR**	**FNMR**	**Acc (balanc.)**	**FMR**	**FNMR**	**Acc (balanc.)**
Flight time		***Self-Detector (No adaptation)***	0.388 (0.014)	**0.180** (0.002)	0.716 (0.007)			
		Self-Detector (Sliding)	0.330 (0.021)	0.205 (0.008)	**0.733** (0.012)	0.292 (0.023)	0.254 (0.011)	0.727 (0.012)
		Self-Detector (Usage Control 2)	**0.186** (0.013)	0.396 (0.012)	0.709 (0.010)			
		Self-Detector (Usage Control R)	0.360 (0.022)	0.188 (0.006)	0.726 (0.012)			
		Self-Detector (Usage Control S)	0.255 (0.017)	0.296 (0.009)	0.725 (0.009)			
	Proposals	*Self-Detector (ETU 0)*	0.483 (0.018)	**0.115** (0.012)	0.701 (0.010)	0.494 (0.018)	**0.118** (0.012)	0.694 (0.010)
	Proposals	*Self-Detector (ETU 1)*	0.441 (0.019)	0.140 (0.014)	0.710 (0.011)	0.448 (0.019)	0.141 (0.014)	0.705 (0.011)
	Proposals	*Self-Detector (ETU 2)*	**0.364** (0.020)	0.171 (0.009)	**0.733** (0.011)	**0.331** (0.021)	0.213 (0.011)	**0.728** (0.012)
nGdv		***Self-Detector (No adaptation)***	0.405 (0.021)	**0.274** (0.008)	0.660 (0.010)			
		Self-Detector (Sliding)	0.284 (0.020)	0.367 (0.009)	**0.674** (0.010)	0.271 (0.018)	0.380 (0.010)	0.675 (0.009)
		Self-Detector (Usage Control 2)	**0.147** (0.013)	0.611 (0.015)	0.621 (0.009)			
		Self-Detector (Usage Control R)	0.312 (0.022)	0.346 (0.008)	0.671 (0.010)			
		Self-Detector (Usage Control S)	0.241 (0.018)	0.443 (0.008)	0.658 (0.009)			
	Proposals	*Self-Detector (ETU 0)*	0.547 (0.020)	**0.153** (0.019)	0.650 (0.012)	0.564 (0.020)	**0.145** (0.017)	0.645 (0.012)
	Proposals	*Self-Detector (ETU 1)*	**0.245** (0.018)	0.394 (0.023)	**0.680** (0.016)	**0.260** (0.017)	0.375 (0.020)	**0.683** (0.015)
	Proposals	*Self-Detector (ETU 2)*	0.364 (0.018)	0.284 (0.013)	0.676 (0.011)	0.349 (0.018)	0.300 (0.011)	0.676 (0.009)
		M2005 (No adaptation)	0.329 (0.035)	0.251 (0.015)	0.710 (0.024)			
		M2005 (DB)	0.251 (0.026)	0.240 (0.014)	0.754 (0.018)			
		M2005 (IDB)	**0.247** (0.023)	**0.190** (0.006)	**0.781** (0.013)			
	Proposals	*M2005 (ETU 0)*	**0.138** (0.020)	0.583 (0.026)	0.640 (0.013)	**0.134** (0.018)	0.614 (0.025)	0.626 (0.011)
	Proposals	*M2005 (ETU 3)*	0.294 (0.016)	**0.177** (0.021)	**0.765** (0.013)	0.247 (0.019)	**0.268** (0.017)	**0.742** (0.014)

Table 7: Results from the statistical test (Wilcoxon Signed Rank Test).

Algorithm	**FMR**	**FNMR**	**Acc (balanc.)**	**FMR**	**FNMR**	**Acc (balanc.)**
	Static baseline			Adaptive baseline (Sliding)		
Self-Detector (ETU 0)	.	+	.	.	+	.
Self-Detector (ETU 1)	.	+	+	.	.	.
Self-Detector (ETU 2)	.	+	+	.	+	.
	Static baseline			Adaptive baseline (IDB)		
M2005 (ETU 0)	+	.	.	+	.	.
M2005 (ETU 3)	.	+	+	.	+	.

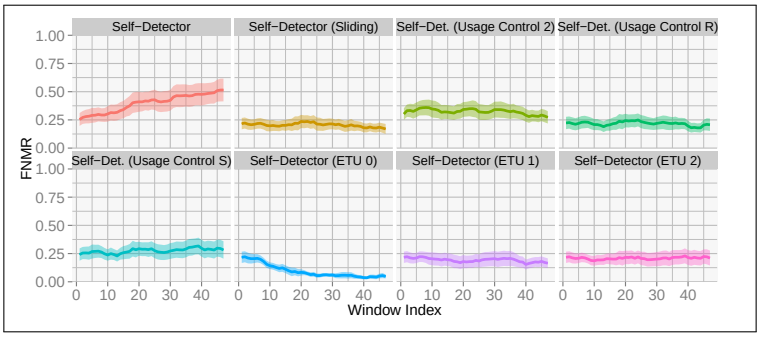
it is better than both the static and adaptive baselines in most cases. It is important to note that a point sign (.) does not mean that ETU is worse, but that it was not statistically better. For balanced accuracy, for example, we have seen that ETU 3 obtained better accuracy than the baselines in CMU, although it was not the better in GREYC-Web dataset.

5.2. Performance Over Time

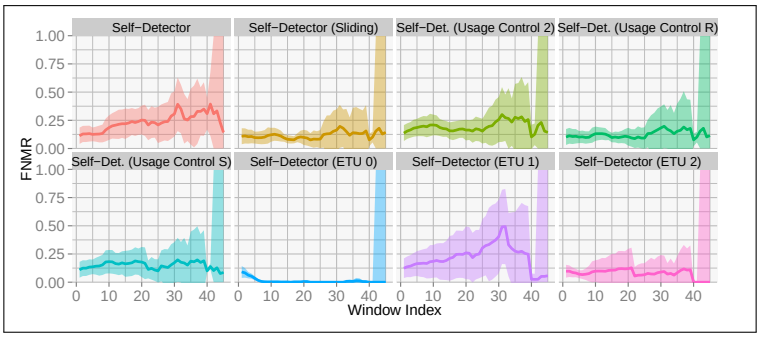
Figures 9, 10, 11 and 12 present the experimental results over time. A description of how these graphs are plotted is shown in Section 4.7. Each line is the average performance at the indicated window index, while the shaded area represents a confidence interval based on standard error. The shaded area shows how the performance varies among the users at the specified evaluation window. Thus, a large shaded area means that the performance presented a high variation among the users.

As expected, static algorithms tend to increase FNMR over time. For *Self-Detector*, adaptive algorithms manage to decrease FNMR over time when compared to their static counterpart (although for GREYC-Web Passwords, which has a short expression, some adaptive algorithms could not decrease it). Thus, adaptive methods showed to be better alternatives than static *Self-Detector* in terms of FNMR.

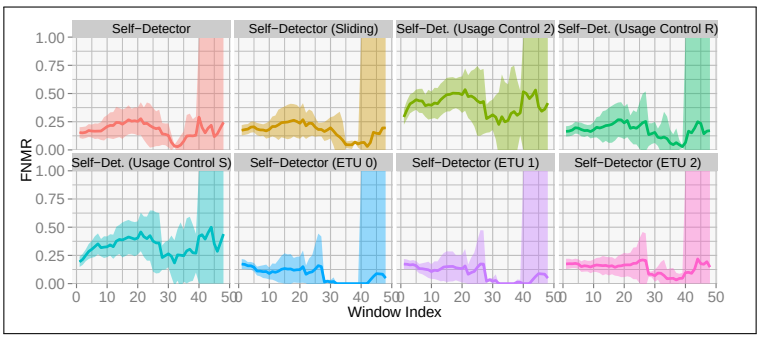
All ETU methods have the same behaviour in the very



(a) CMU



(b) GREYC-Web (logins)



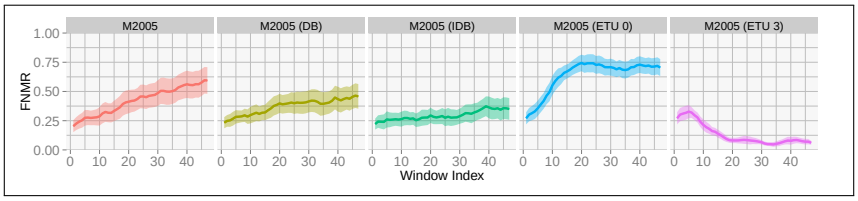
(c) GREYC-Web (passwords)

Figure 9: Self-Detector - False non-match rate (FNMR) over time.

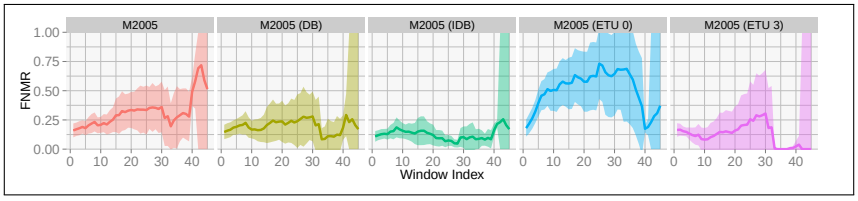
first moments. This happens because they work as a simple *Sliding* method before reaching the minimum amount of samples for using the rules involving the negative gallery. Note that in GREYC-Web, some users have few samples, resulting in short data streams. It must be reminded that a lower value was used for negative ETU activation in GREYC-Web. Hence, the *negative procedure* can be activated earlier than in the experiments on the CMU dataset.

Overall, ETU methods for *Self-Detector* were better than *Sliding* in terms of FNMR. In some cases, the performance difference was small. With regard to M2005, there was a decrease in FNMR over time with ETU 3 (it is clearer on CMU). This is a good result as a recent study has shown that adaptive M2005 methods tend to increase FNMR over time, although in a lower rate than static M2005 [9] (Figure 10 for CMU illustrates this behaviour). ETU 3 initially increased the FNMR for the CMU dataset, until the *negative procedure* was activated (when the minimum amount of negative samples was obtained).

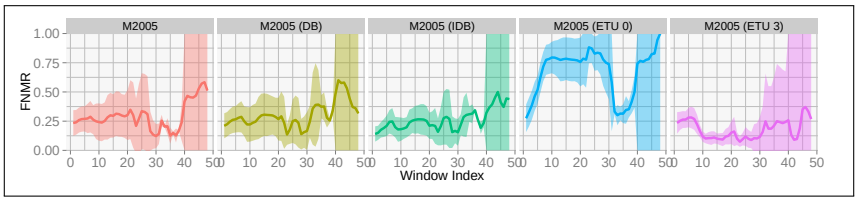
Still regarding ETU 3 in CMU, the improved plot shows that the deviation of FNMR performance among all users



(a) CMU



(b) GREYC-Web (logins)



(c) GREYC-Web (passwords)

Figure 10: M2005 - False non-match rate (FNMR) over time.

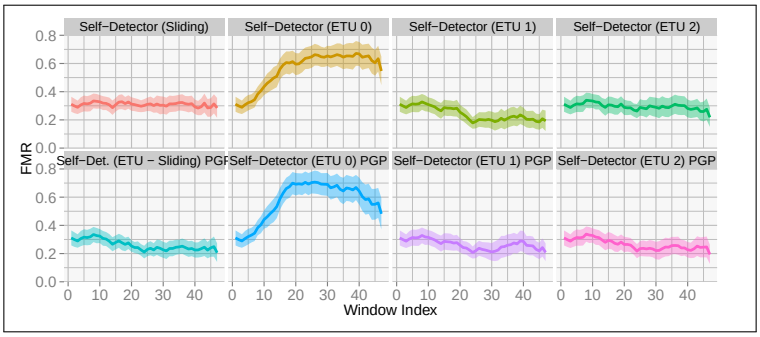
was lower than for the other M2005-based algorithms, as the shaded area shows. It is a good result, since all users have the same amount of samples in this dataset, so the decreased variation is not due to a change in the number of considered users. On GREYC-Web datasets, for example, some users have more samples than others and, therefore, their data streams have different sizes. Thus, in later windows of the analysis, some users are not considered. In the very last part of these graphs over time only one user remained in the end (only for GREYC-Web as CMU has the same amount of samples per user). In this case, we considered a hypothetical variation over the full range $[0; 1]$, which is the reason why the graph suddenly increases to a constant very high standard error in the very last part of GREYC-Web plots.

Figures 9 and 10 show the FNMR for all algorithms, as the main goal of the algorithms evaluated here is to reduce FNMR (i.e. the user model should be closer to the current user behaviour). However, for FMR, only the graphs for the algorithms that support PGP are shown in Figures 11 and 12. This allows to evaluate the effect of ETU PGP over time (PGP main goal is to reduce FMR).

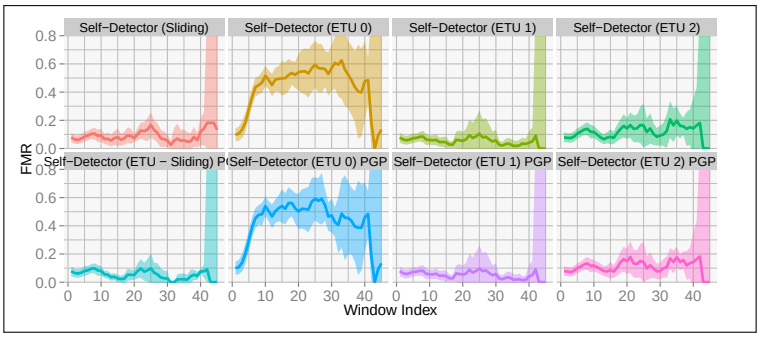
In terms of FMR, the behaviour of the algorithms did not change much over time when PGP was applied. The usage of a reduced value for activating the negative procedure in the GREYC-Web dataset may have negatively affected the performance of PGP, since it creates an imbalance between the positive and negative galleries. However, it is possible to see that the FMR was reduced in some cases. For *Self-Detector*, it mainly occurs on datasets CMU (ETU 0) and GREYC-Web Passwords (*Sliding* and ETU 2). M2005-based algorithms improved FMR for ETU 3 on all datasets (this improvement on FMR was also observed in the last section on the overall results).

6. Conclusion

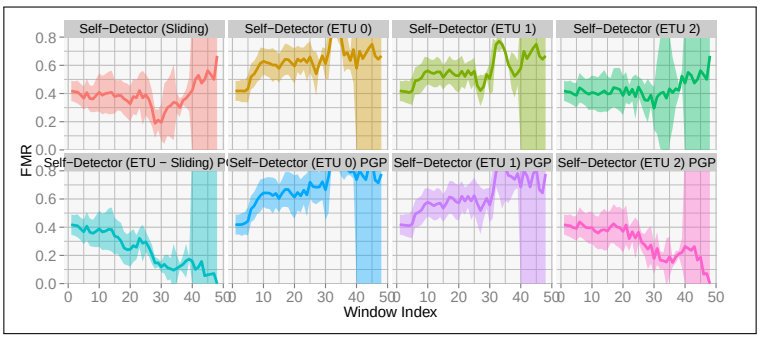
Some recent studies on biometrics have shown the importance to adapt the user model/template using template



(a) CMU



(b) GREYC-Web (logins)

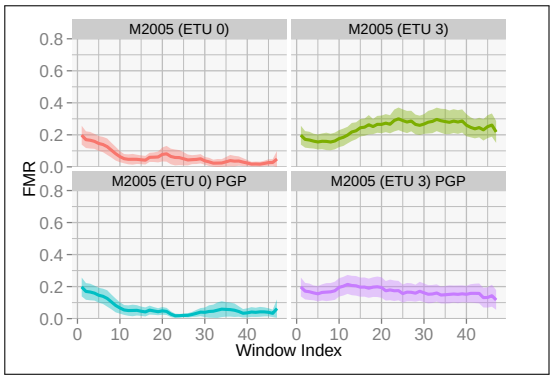


(c) GREYC-Web (passwords)

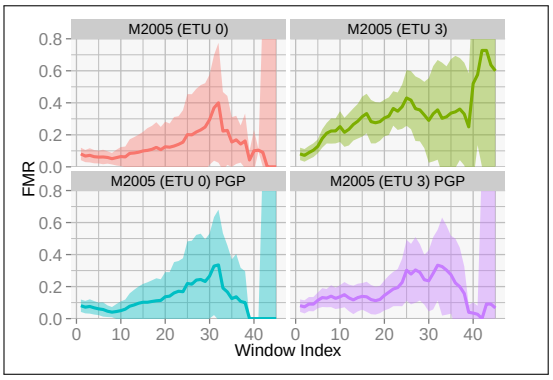
Figure 11: Self-Detector - False match rate (FMR) over time - PGP.

update/adaptive biometric systems. Keystroke dynamics is an important biometric modality which has been reported to undergo changes over time. Most of current work in the area only makes use of samples classified as genuine/positive to adapt the user model. This paper investigated several strategies to use all available samples, including those classified as impostor/negative. It is named as *Enhanced Template Update* (ETU).

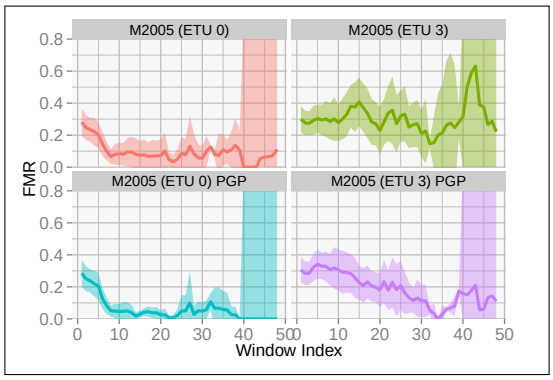
ETU methods work mainly to change the classification decision, reducing FNMR. In addition, the Positive Gallery Protection (PGP), which is also part of the ETU framework, was specifically designed to avoid errors during template update, reducing FMR. From the results obtained in this paper, we conclude that ETU approaches have competitive performance, although may not the best solution in all cases. For the CMU dataset, which is the largest one, ETU 3 was the overall best algorithm regarding balanced accuracy. However, for the other shorter datasets, current adaptive algorithms performed better. It also indicates the need for additional research on the combination of positive and negative galleries. An investigation of the proposed approach on other datasets (which



(a) CMU



(b) GREYC-Web (logins)



(c) GREYC-Web (passwords)

Figure 12: M2005 - False match rate (FMR) over time - PGP.

do not exist today) would clarify this point. The paper has also shown that different feature vectors can impact the predictive performance, but the tendencies of improvement for the adaptive methods are similar among them.

Moreover, the current study contributed by showing improvements on visualization tools for analyzing the behaviour of algorithms over time. An interesting result from the graphs produced by these tools is that ETU 3 has a reduced FNMR performance variation among all users in CMU (illustrated by the reduced shaded area). Furthermore, it was possible to see the performance over time for the whole data stream of all users in the GREYC-Web dataset.

This study proposed the use of both positive and negative samples to support template update. Several aspects of performance evaluation of template update have been discussed too. We expect that this paper can lead to further studies in the area, mainly to expand the idea of using negative samples to support adaptation. In line with this, we highlight some aspects for future studies. The usage of another mechanism to avoid the inclusion of weakly classified samples could be investigated. Other strategies to combine positive and negative data under our framework can also be proposed. The proposed *Enhanced Template Update* framework can also be evaluated on other biometric modalities.

Acknowledgment

The authors would like to thank LaBRI/Université de Bordeaux for the financial support to the Enhanced Template Update project. We also would like to thank Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) and Fundação de Amparo

à Pesquisa do Estado de São Paulo (FAPESP - processes 2012/25032-0, 2012/22608-8 and 2013/07375-0).

References

[1] D. Hosseinzadeh, S. Krishnan, Gaussian mixture modeling of keystroke patterns for biometric applications, Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Trans. on 38 (6) (2008) 816–826.

[2] A. Peacock, X. Ke, M. Wilkerson, Typing patterns: a key to user identification, Security Privacy, IEEE 2 (5) (2004) 40–47.

[3] R. Giot, C. Rosenberger, B. Dorizzi, Performance evaluation of biometric template update, CoRR abs/1203.1502.

[4] F. Roli, L. Didaci, G. Marcialis, Adaptive biometric systems that can improve with use, in: N. Ratha, V. Govindaraju (Eds.), Advances in Biometrics, Springer London, 2008, pp. 447–471.

[5] N. Poh, A. Rattani, F. Roli, Critical analysis of adaptive biometric systems, Biometrics, IET 1 (4) (2012) 179–187. doi:10.1049/iet-bmt.2012.0019.

[6] R. Giot, M. El-Abed, C. Rosenberger, Web-based benchmark for keystroke dynamics biometric systems: A statistical analysis, in: Int. Conf. on Intelligent Information Hiding and Multimedia Signal Processing, 2012, pp. 11–15.

[7] A. Rattani, G. Marcialis, F. Roli, Temporal analysis of biometric template update procedures in uncontrolled environment, in: G. Maino, G. Foresti (Eds.), Image Analysis and Processing - ICIAP 2011, Vol. 6978 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2011, pp. 595–604. doi:10.1007/978-3-642-24085-0_61.

[8] R. Giot, C. Rosenberger, B. Dorizzi, Hybrid template update system for unimodal biometric systems, in: IEEE BTAS 2012, IEEE, 2012, pp. 1–7.

[9] P. H. Pisani, A. C. Lorena, A. C. de Carvalho, Adaptive positive selection for keystroke dynamics, Journal of Intelligent & Robotic Systems (2014) 1–17doi:10.1007/s10846-014-0148-0.

[10] P. Kang, S.-s. Hwang, S. Cho, Continual retraining of keystroke dynamics based authenticator, in: S.-W. Lee, S. Li (Eds.), Advances in Biometrics, Vol. 4642 of LNCS, Springer Berlin / Heidelberg, 2007, pp. 1203–1211.

[11] A. Messerman, T. Mustafic, S. Camtepe, S. Albayrak, Continuous and non-intrusive identity verification in real-time environments based on free-text keystroke dynamics, in: Biometrics (IJCB), 2011 Int. Joint Conf. on, 2011, pp. 1–8.

[12] S. T. Magalhaes, K. Revett, H. M. D. Santos, Password secured sites: Stepping forward with keystroke dynamics, in: Proceedings of the International Conference on Next Generation Web Services Practices, NWESP '05, IEEE Computer Society, 2005, pp. 293–. doi:10.1109/NWESP.2005.62.

[13] P. H. Pisani, A. C. Lorena, A. C. Ponce de Leon Carvalho, Adaptive approaches for keystroke dynamics, in: Neural Networks (IJCNN), The 2013 International Joint Conference on, 2015, pp. 1–8. doi:10.1109/IJCNN.2015.7280467.

[14] T. Stibor, J. Timmis, Is negative selection appropriate for anomaly detection, ACM GECCO (2005) 321–328.

[15] T. Scheidat, A. Makrushin, C. Vielhauer, Automatic template update strategies for biometrics, Tech. rep., Otto-von-Guericke University of Magdeburg, Germany (2007).

[16] P. H. Pisani, A. C. Lorena, A. C. Ponce de Leon Carvalho, Adaptive algorithms in accelerometer biometrics, in: Intelligent Systems (BRACIS), 2014 Brazilian Conference on, 2014, pp. 336–341. doi:10.1109/BRACIS.2014.67.

[17] H. joo Lee, S. Cho, Retraining a keystroke dynamics-based authenticator with impostor patterns, Computers & Security 26 (4) (2007) 300–310.

[18] D. Arthur, S. Vassilvitskii, K-means++: The advantages of careful seeding, in: Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA '07, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2007, pp. 1027–1035.

[19] C. Bishop, Pattern Recognition and Machine Learning, Springer, 2006.

[20] M. C. Naldi, R. J. G. B. Campello, E. R. Hruschka, A. C. P. L. F. Carvalho, Efficiency issues of evolutionary k-means, Appl. Soft Comput. 11 (2) (2011) 1938–1952. doi:10.1016/j.asoc.2010.06.010.

[21] K. Killourhy, R. Maxion, Why did my detector do that?! predicting keystroke-dynamics error rates, in: Recent Advances in Intrusion Detection, Vol. 6307 of LNCS, Springer, 2010, pp. 256–276.

[22] R. Giot, M. El-Abed, C. Rosenberger, Greyc keystroke: a benchmark for keystroke dynamics biometric systems, in: IEEE BTAS 2009, IEEE, 2009, pp. 419–424.

[23] P. H. Pisani, A. C. Lorena, Emphasizing typing signature in keystroke dynamics using immune algorithms, Applied Soft Computing 34 (2015) 178 – 193. doi:http://dx.doi.org/10.1016/j.asoc.2015.05.008.

[24] K. Xi, Y. Tang, J. Hu, Correlation keystroke verification scheme for user access control in cloud computing environment, The Computer Journal 54 (10) (2011) 1632–1644. doi:10.1093/comjnl/bxr064.

[25] P. S. Teh, A. B. J. Teoh, S. Yue, A survey of keystroke dynamics biometrics, The Scientific World Journal (2013) 1–24doi:10.1155/2013/408280.

[26] P. H. Pisani, A. C. Lorena, A systematic review on keystroke dynamics, Journal of the Brazilian Computer Society 19 (4) (2013) 573–587.

[27] Commons Math: The Apache Commons Mathematics Library (3.3), http://commons.apache.org/proper/commons-math/ (2014).

[28] J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. (2006) 1–30.

[29] S. Holm, A simple sequentially rejective multiple test procedure, Scandinavian Journal of Statistics 6 (1979) 65–70.